ESD-TR-84-294

2

Semiannual Technical Summary

Restructurable VLSI Program

30 September 1984

Lincoln Laboratory

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Lexington, Massachusetts

Prepared for the Defense Advanced Research Projects Agency under Electronic Systems Division Contract F19628-85-C-0002.

ESD-TR-84-294

2

Semiannual Technical Summary

AD-A152 340

Restructurable VLSI Program

30 September 1984

Lincoln Laboratory

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

LEXINGTON, MASSACHUSETTS

Prepared for the Defense Advanced Research Projects Agency
under Electronic Systems Division Contract F19628-85-C-0002.

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
LINCOLN LABORATORY

RESTRUCTURABLE VLSI PROGRAM

SEMIANNUAL TECHNICAL SUMMARY REPORT
TO THE
DEFENSE ADVANCED RESEARCH PROJECTS AGENCY

1 APRIL — 30 SEPTEMBER 1984

ISSUED 5 FEBRUARY 1985

LEXINGTON MASSACHUSETTS

ABSTRACT

This report describes work performed on the Restructurable VLSI Program sponsored by the Information Processing Techniques Office of the Defense Advanced Research Projects Agency during the period 1 April through 30 September 1984.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

RESTRUCTURABLE VLSI PROGRAM

I. PROGRAM OVERVIEW AND SUMMARY

A. OVERVIEW

The main objective of the Lincoln Restructurable VLSI Program (RVLSI) is to develop design methodologies, architectures, design aids, and testing strategies for implementing wafer-scale systems with complexities approaching a million gates. In our approach, we envisage a modular style of architecture comprising an array of cells embedded in a regular interconnection. Ideally, the cells should consist of only a few basic types. The interconnection matrix is a fixed pattern of metal lines augmented by a complement of programmable switches or links. Conceptually, the links could be either volatile or nonvolatile. They could be of an electronic nature, such as a transistor switch, or could be permanently programmed through some mechanism such as a laser. The RVLSI Program is currently focusing on laser-formed interconnect.

The link concept offers the potential for a highly flexible, restructurable type of interconnect technology that could be exploited in a variety of ways. For example, logical cells or subsystems found to be faulty at wafer-probe time could be permanently excised from the rest of the wafer. The flexible interconnect could also be used to circumvent faulty logic and tie in redundant cells judiciously scattered around the wafer for this purpose. Also, the interconnect could be tailored to a specific application in order to minimize electrical degradations and performance penalties caused by unused wiring and links.

Further, the testing of a particular logical subsystem buried deep within a complex wafer-scale system poses a very difficult problem. A properly designed restructurable interconnect matrix could be temporarily configured to improve both the controllability and observability of internal cells from the wafer periphery. In this way, each component cell or a manageable cluster of cells could be tested in a straightforward manner using standard techniques. With an electronic linking mechanism, it is possible to think in terms of a dynamically reconfigurable system. Such a feature could be used to alter the functional mode of a system to changes in the operating scenario, or it could be used to support some degree of fault tolerance if the system architecture was suitably designed.

Several major areas of research have been identified in the context of the RVLSI concept:

(1) System architectures and partitioning for whole-wafer implementation.

(2) Placement and routing strategies for optimal utilization of redundant resources and efficient interconnect.

(3) Assignment and linking algorithms to exploit redundancy and flexible interconnect.

(4) Methods for expediting cell design with emphasis on functional level descriptions, enhanced testability, and fault tolerance.

(5) Methods for testing complex, multiple-cell, whole-wafer systems.

Complementary work on the development of various link and interconnect technologies as well as fabrication/processing technology is being supported by the Lincoln Air Force Line Program, and results are reported under the Lincoln Laboratory Advanced Electronic Technology Quarterly Technical Summary.

B. SUMMARY OF PROGRESS

Work for this period is reported under three headings: Design Aids for RVLSI (Section II), Applications (Section III), and Testing (Section IV). The following paragraphs summarize progress to date.

1. Design Aids for RVLSI

The MacPitts silicon compiler continues to find new applications. Several new chips using MacPitts designs were fabricated in the MOSIS facility. Copies of the MacPitts program have been distributed to four more DARPA contractors and government agencies.

An experiment was done applying the simulated annealing technique to the problem of optimally ordering the columns of a Weinberger array. The purpose was to see if the simulated annealing method could provide better area efficiency results than the min-cut approach now used in the Lincoln Boolean Synthesizer (LBS). The results were generally negative. The running times of the algorithm were longer and the resulting designs were more complicated.

Several improvements have been added to the wafer design and fabrication systems. An improved method for testing the wafer interconnect has been developed. The method uses a more sophisticated modeling technique for calculating the expected capacitance of a track directly from the wafer description files. The technique allows the capacitance measurements to be used for testing even for small groups of tracks. Improvements also have been made in the laser linking process. The laser restructuring and optical probe testing machinery have been integrated so that these operations can be performed together without manual intervention.

A DSP-oriented silicon compiler is being developed for generating layouts of digital circuits directly from a high-level signal-processing language. The language design is complete and a compiler has been written to parse the language into a connection list for a set of basic signal-processing primatives. Design of a set of such primatives, including a multiplier, adder, subtractor, and delay elements has begun. A basic floor plan and layout scheme have been developed for the resulting layout.

2. Applications

Work continues on the development of a wafer-scale implementation of the Myers-Rabiner Dynamic Time Warping algorithm for connected-word speech recognition. A new architecture which better fits the constraints of the wafer system has been developed. An extensive simulation has been written allowing the recognition performance of the system to be evaluated for different design parameters such as path constraint, distance metric, and parameter bit-quantization. The simulation has enabled us to make the decisions necessary to proceed with the detailed circuit design.

3. Testing

Preparations are being made to distribute the optical probe to DARPA contractors. Fabrication of the necessary machined parts necessary for the optical probe bit is under way. A manual has been prepared containing details of the probe construction and its application.

II. DESIGN AIDS FOR RVLSI

A. MACPITTS STATUS

Three chips simulating the behavior of auditory neurons were designed using MacPitts, manufactured in MOSIS 4-μm NMOS, and tested. The chips range from 3000 to 3600 transistors, making them the most complex chips designed using MacPitts that have been successfully manufactured. Of the 25 chips received, 14 functioned correctly at clock rates around 1.25 MHz. A detailed report on the testing results was sent to MOSIS.

B. LINCOLN BOOLEAN SYNTHESIZER (LBS)

A program to order the columns of Weinberger arrays based on simulated annealing[1] was written. Our interest was to compare the results of the annealing process to the present recursive min-cut heuristic technique currently in use in the LBS. The abstract geometric configuration modeling the problem can be described as follows. A set of vertical lines called columns, whose number is fixed, must be placed along a straight line. A set of nets, interconnecting the cells, must be laid out in horizontal tracks so that no two nets intersect. The columns are to be reordered so that the number of horizontal tracks employed is minimal, or so that the total net length is minimal. Note that these two objective functions are closely related, even if we cannot write an explicit mathematical formula for the dependence between the two quantities.

The min-cut approach[2] tries to divide the columns into two sets, each containing approximately the same number of columns, in a way such that the number of nets connecting columns belonging to both sets is minimized. This number is a lower bound for the number of tracks necessary to lay out the nets. The procedure is applied recursively to each subset of the columns.

In the simulated annealing procedure, every possible configuration has a cost, called its energy, and we try to exchange positions between columns so that this energy decreases. The exchanges are determined using probabilistic selection and acceptance procedures. For our case, the energy of a configuration was defined as proportional to the total horizontal wire length in the array. Two selection procedures were tried: arbitrary exchanges between any two columns in the array, and exchanges only between close neighbors. The data structures employed allowed quick calculation of the energy variation after an exchange, avoiding the time-consuming total calculation of the energy.

The comparisons were performed using a set of randomly generated examples. The number of columns varied between 25 and 100, and the number of columns per net was obtained using two different probability distributions — uniform and Poisson.

The comparisons were not favorable to the annealing algorithm. While these results should not be taken as final since the tests performed cannot be considered exhaustive, running times for the annealing algorithm were considerably longer, and the final area required for the layout was larger than that obtained using the min-cut algorithm. This happened even when a large number of trials were allowed in the annealing procedure (about 100,000, even though for the case of 100 columns this represents a totally negligible percentage of the 100! possible configurations). One interesting use for the annealing procedure was discovered where it was applied to a starting configuration obtained by the min-cut procedure. This does not improve the number of required tracks, but reduces the total net length.

C. WAFER DESIGN AND FABRICATION TOOLS

Improvements have been made in two areas of the CAD software used for design, test, and linking of the wafer circuits. Improved procedures for testing the wafer interconnect lines have been developed and the wafer restructuring process has been further automated.

1. Interconnect Measurement and Analysis

In restructuring a wafer, it is necessary to know which tracks and stubs are good. We determine this by measuring the capacitance to the substrate at each circuit and track pad and analyzing the data with the LSH program "analyze." This process has been improved by adding automatic location of the pads on each track, and by incorporating an improved analysis technique for assessing the quality of a track.

A new test generation method has been developed which automatically extracts the locations of pads to be probed from the LSH wafer and cell description files. The file of locations drives a new tester program which is usable for any wafer layout. New formatting programs have been written which list capacitance data by pad name instead of grid locations, thus making the data easier to understand by users. This new system eliminates the need for manual extraction of the probe pads and makes the testing procedure independent of wafer design.

The quality of track is determined by measurement of the capacitance of each track. Testing results are input to the "analyze" program which determines the position and length of each track and the number and position of links on the tracks from LSH wafer description files.

An improved analysis program has been written. It allows the user to specify an initial guess for the model parameters which permits a better pruning of bad tracks and, therefore, better analysis of small groups of tracks. This guess could come from knowledge of the wafer characteristics or from the computed values of another group of easily analyzed tracks.

The data from all lines with capacitance within user-specified limits are used in a least-squares fit with up to seven parameters. The first three parameters are a constant capacitance, capacitance per unit track length, and capacitance per link. The next four parameters are the variation of track and link capacitance along and perpendicular to the tracks.

Experience has shown that some care must be used to group lines together, for instance, to separate out adjacency effects. From the derived parameters, the capacitance for each line is calculated and tracks with measured values different by more than a user-specified amount are marked bad. This new program was successful on a small group of tracks on a FFT wafer which the old program was not able to analyze correctly. We are also considering variations on the capacitance model and on methods for extracting model parameters from the least-squares fit of measured data, methods for handling other sources of capacitance (power bus crossovers, pads, circuits), and tracks which include both horizontal and vertical segments. These more complex tracks could be the result of partial linking or wafer layouts which are more complex than the Integrator and FFT circuits with which we have been working.

2. LSH and RWED Automation

Previously described improvements in the Linking Shell (LSH) and the Restructurable Wafer Editor (RWED, the control program for the restructuring process) for increased automation have been used in restructuring Integrator and FFT wafers. Restructuring and laser-probe testing of the interconnect can now be done without manual intervention. The computer can control the table position, the laser power and shutter, and electrical connections to the wafer circuit. The computer can also sense the current in the wafer when it is probed. The next step will be to integrate functional testing into this procedure.

D. DSP-ORIENTED SILICON COMPILER

The DSP Silicon Compiler is a program for automatically generating custom VLSI designs for signal-processing applications. The program will convert the circuit specification, written in a high-level signal-processing language, directly into CIF which can be used for circuit fabrication. Several tasks are proceeding in parallel: the language design, the compiler design, the resource allocation, the cell design, and the final layout.

The compiler will generate designs using as its basic cells signal-processing primatives such as multipliers, adders, and delay elements. Bit serial arithmetic will be used to minimize the problem of interconnection of the basic cells and to produce the designs compatible with the restructurable wafer technology.

A first step toward the realization of a digital signal-processing-oriented silicon compiler has been the design of a high-level, digital filter design language. The language which is based on "C," is capable of specifying arbitrary digital filters of the type described by finite difference equations. The filter designer can control both the structure and the computational resources (adders and multipliers) used in a design by means of explicit and implicit language constructs. Resource multiplexing and state memory management are arranged for by the compiler without any explicit user intervention. The compiler will also automatically generate all necessary control and clock signals. Arithmetic operations on the chip are done serially in the interest of minimizing silicon area requirements; however, provisions for

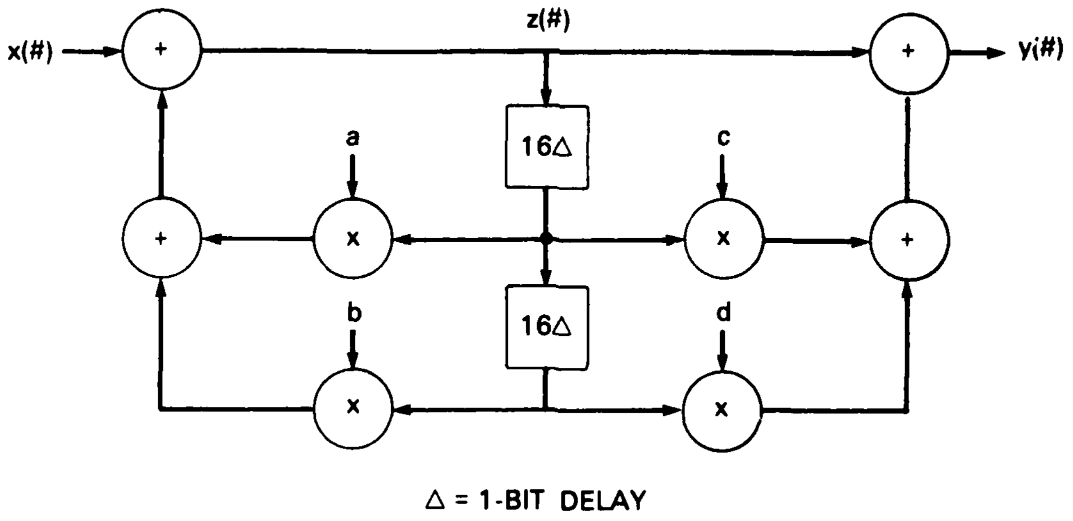
```
wordsize 16;
input x{1} ;
output y{6} ;
coefsize 12;
coef a = .25 , b = .75 ;
coef c = 1.25 , d = .5 ;
var z;

%

par {
        z(#) = x(#) + (a*z(#-1) + b*z(#-2)) ;
        y(#) = z(#) + (c*z(#-1) + d*z(#-2)) ;
}
```

147987-N

Figure 1. Second-order section specification.



147988-N

Figure 2. Second-order section.

parallel data input and output from the filter chip will be provided. The filter design language will also handle up-and-down scaling, rectification, and decimation in time.

Figure 1 shows an example of the specification language applied to the problem of realizing a second-order filter section in canonic form. The first part of the description specifies the data word size and filter coefficient word size. Pins 1 and 6 are declared as input and output and labeled x and y, respectively. An internal variable called z is declared. The remainder of the program uses the language of ordinary algebra to describe the signal processing that is to take place. In these equations, the symbol # is used to indicate the time variable. The special construct par {...} is used to indicate that it is the designer's desire to have all arithmetic applications in the enclosing curly brackets performed in parallel. As a consequence, this example will require four multipliers and four adders for its realization in silicon. Figure 2 is a block diagram of the filter structure which the compiler will fabricate when presented with the specification of Figure 1.

As a further example of the language, consider a specification that is identical to that given in Figure 1 except without the par {...} statement. This construction signals the designer's intent that the two lines of computation be performed sequentially in time. The computation on any single line is still to take place in parallel. This allows the multiplier, an expensive resource in silicon, to be shared among several equations, if the computation rates permit. This modified specification requires only two multipliers and two adders, but requires a more complicated control structure to multiplex the computational resources. Figure 3 depicts the filter structure that will be derived by the compiler for the modified specification. The input and output buffers are required in this case because the clock rate of the internal circuitry must now be twice the clock rate used for the input and output data streams in order to accommodate the sequential computation.

A parser for the filter design language just described has been written and debugged. The parser accepts the source code of Figure 1 as input and produces a series of data structures called parse trees. These parse trees preserve all the information contained in the source code but are arranged in a way to facilitate further processing for the purpose of extricating information related to computational resource and multiplexing requirements, connectivity, and clock rates.

The parser is a "C" program that was written with the aid of the software development tools LEX and YACC. LEX is a Lexical Analyzer Generator which allows the programmer to specify the tokens (names, operators, numbers, etc.) of the source language at a fairly high level. LEX then generates a "C" program that breaks up the input source text into a stream of tokens suitable for further processing. YACC allows the programmer to give a high-level description of the intertoken structure or syntax of the source language to be parsed. Given such a description, YACC produces a "C" program that "recognizes" the various grammatical constructs of the language and then invokes the appropriate user-defined actions. In the present case, these actions construct the parse trees that embody the structure of the source code statements.

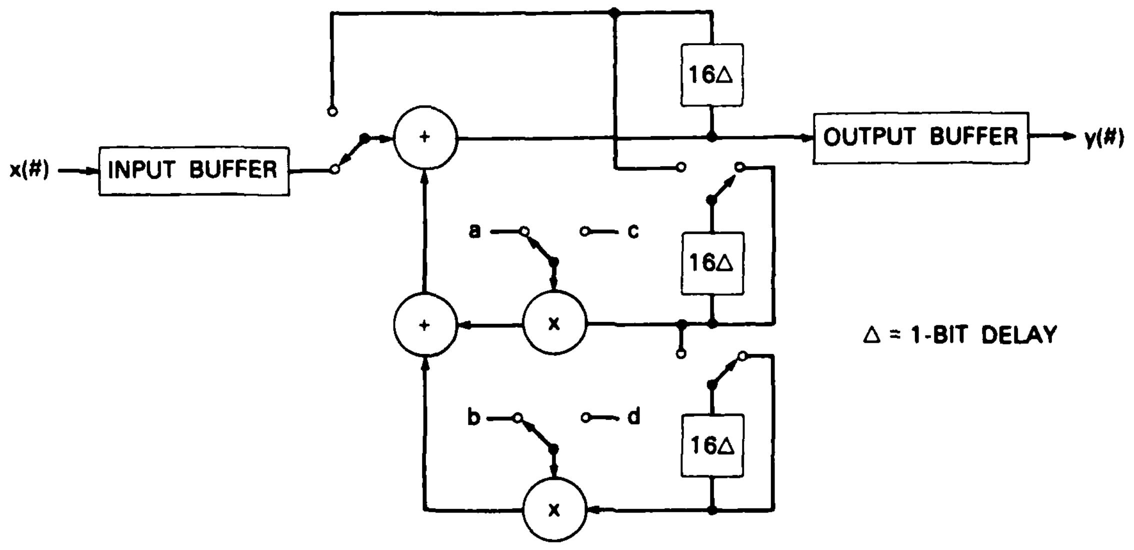


Figure 3. Multiplexed second-order section.

A set of programs which convert the parse trees to a netlist is nearing completion. These programs provide multiplexing of computational resources and memory. A pattern-matching algorithm allocates resources in order to minimize multiplexing and interconnection. As indicated above, Figures 2 and 3 illustrate the need for multiplexing of computational resources when the computation takes place sequentially in time. In Figure 3, multiplexing is implemented by the two position switches shown. Multiplexing of resources in sequential calculations gives rise, in turn, to a need for careful resource assignment.

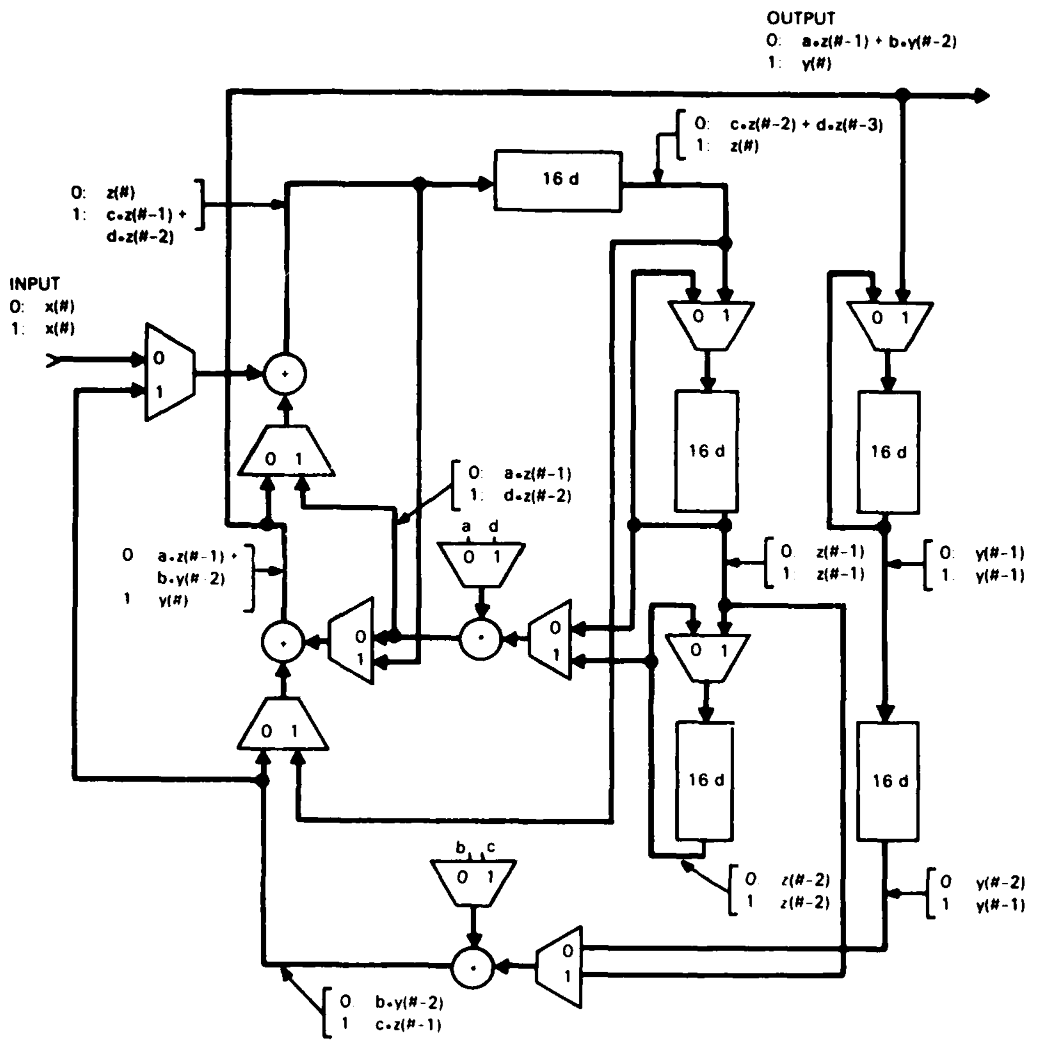
Figure 4 shows a circuit which could be derived from the following filter equations:

$$z(\#) = x(\#) + a * z(\# - 1) + b * y(\# - 2)$$

$$y(\#) = z(\#) + c * z(\# - 1) + d * z(\# - 2) \quad .$$

In particular, the interconnect of Figure 4 illustrates the worst-case random resource assignment for the filter. In contrast, Figure 5 demonstrates the results of resource assignment using the pattern-matching algorithm. Note the reductions in the number of multiplexers, interconnect, and memory. For each type of resource (adder, multiplier, delay, etc.) the algorithm determines a reference pattern of interconnection based on the input statement which makes most use of that type of resource. Resources of this type in all other statements are then allocated such that the pattern created by the allocation best matches the reference pattern. The result is the disappearance of multiplexers within groups of resources when a given group appears in several statements with the same connection pattern.

The clocking and control signal requirements of the filters described by the design language have been studied in detail. The functionality for most of the basic units needed



11 MULTIPLEXERS
5 MEMORY UNITS

Figure 4. Worst-case resource assignment.

147990-N

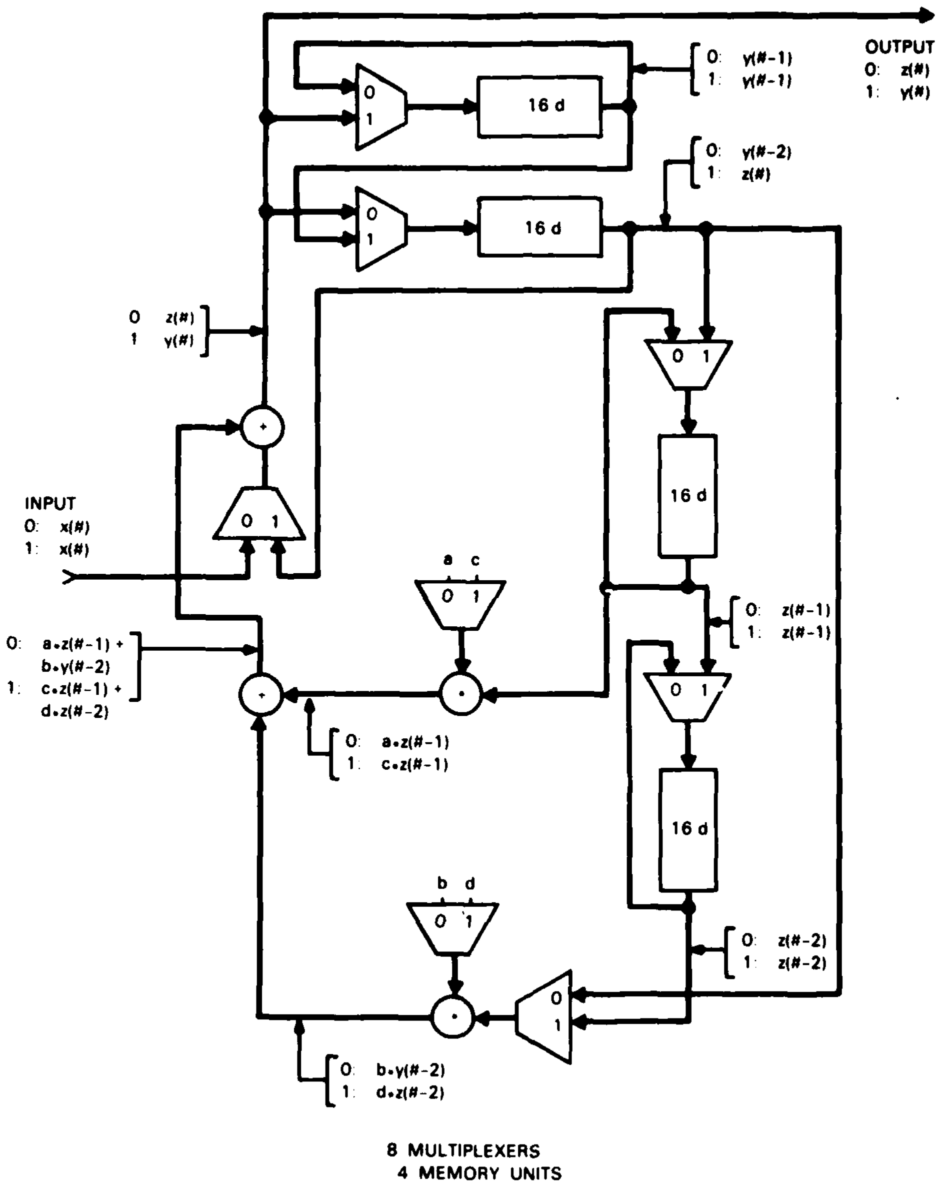


Figure 5. Resource assignment using pattern matching.

to control the computation flow in the chosen pipelined, bit serial architecture has been determined. The next stage will be to identify the specific CMOS building blocks to be used for the implementation of the clock and control signal-generating circuitry. Programs to translate the control requirements of any specific filter into CMOS library items can then be written.

The study of the basic hardware blocks needed to realize digital filters was started. The serial multiplier has received particular attention because it is the most complex block needed and its realization is area intensive. There are several variations of the basic design described by Lyon,[3] differing mainly in the data types they operate on and their latencies. Our focus has been toward the efficient partitioning of the designs into small subunits such that paramaterized designs can be automatically laid out and interconnected. The compiler will be able to form multipliers which use either fractional data or integer/fractional coefficients.

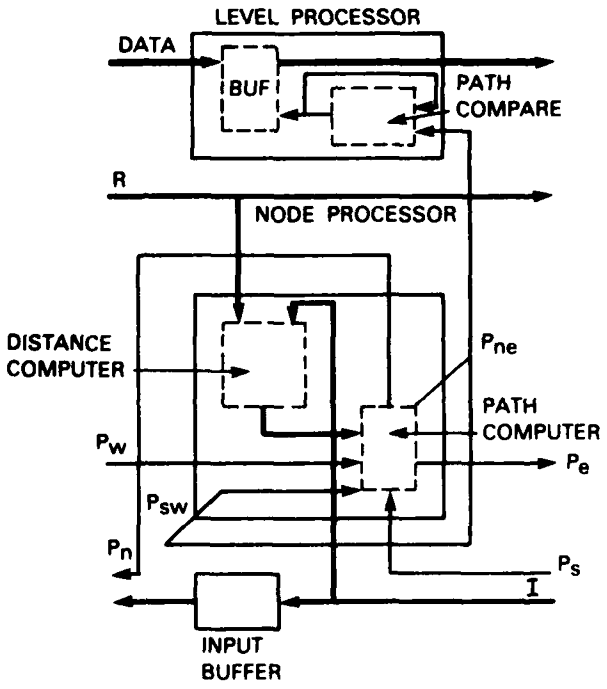
Multiplier units using bit-serial arithmetic have been designed in previous RVLSI projects.[4] The design consists of five basic subunits which may be combined in various ways to generate bit-serial two's complement multipliers, which differ in the coefficient length and radix point. The existing layouts use the design rules for the internal Lincoln Laboratory 5-μm CMOS process. While they cannot be used directly for this project, they have served as important guides in obtaining area and time estimates, and insights for the future layout of the silicon compiler units. Similar descriptions exist as well for a bit-serial adder and subtractor.

III. APPLICATIONS

A. DTW SYSTEM

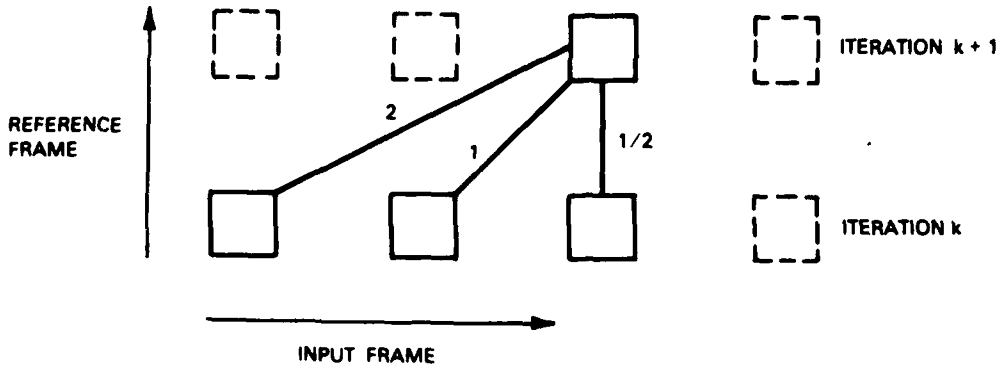
1. Wafer Architecture

A wafer-scale implementation of a Dynamic Time Warping (DTW) array processor is being developed to implement the Myers-Rabiner DTW level-building algorithm for connected-word recognition.[5] Based on the results of work at the University of California at Berkeley,[6] the decision was made to concentrate on filter bank front-end processing for the wafer. We have extended their results through bit-level simulations and architectural analysis aimed at the wafer application. These studies have led to an architecture with high throughput and well matched to the constraints of the restructurable wafer technology. The new processing element is composed of three cell types: a Node Processor, a Level Processor, and an Input Buffer (Figure 6). The Node Processor includes both the distance and path computers.



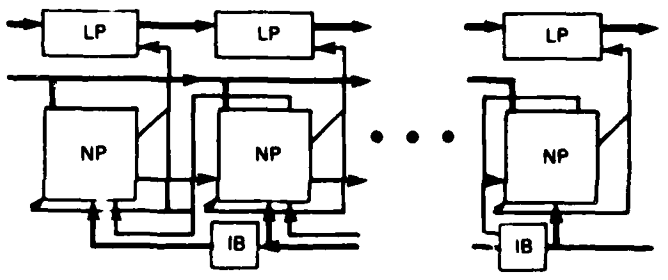
147992-N

Figure 6. Processing element.



147993-N

Figure 7. "Flipped" Itakura path constraint showing data flow between node processors on successive iterations of the DTW algorithm.



LP LEVEL PROCESSOR
NP NODE PROCESSOR
IB INPUT BUFFER

147994-N

Figure 8. Row architecture.

The distance computer calculates a vector distance between the input and reference speech frames. This calculation will be an accumulated sum of magnitude differences (Chebyshev Norm) or sum of squared differences (squared Euclidean distance). The distance computer contains a parallel subtractor followed by a table lookup of either the magnitude or the square of the difference. This result will enter a word parallel adder-accumulator loop. Two accumulator registers will be performing either the accumulation of the current distance calculation or be serially transferring the previous distance to the path computer.

The path computer performs the operation of selecting the best path to this node by choosing the minimum distance entering through the three nodes from the previous row, as shown in Figure 7. It then computes the new path value by adding the distance measure calculated at this node to the selected path value. The local distance is scaled by the weighting factors of 1/2, 1, or 2, or depending on the path choice, before being added to the minimum accumulated path distance. This sum is passed to the next row of processing nodes according to the path constraint selected.

A row-oriented architecture (Figure 8) was selected for the wafer implementation. In the row architecture, the row of nodal elements processes one frame of a reference template against a row of input frames on each iteration. On the next iteration, the next reference frame is processed and a new input frame is shifted into the end processor.

At the end of the reference template, the path calculations are transferred to the level processors. The Level Processor eliminates most of the wafer/host data transfers by making the inter-reference-word level decisions on the wafer. At the end of each reference word the accumulated paths, which appear out of the last iteration of the node processing, are compared with the previously selected minimum paths resulting from all the earlier reference-word comparisons. This decision is a simple minimum selection which can be done with the same type of minimizer used in the path computer. The buffer in the Level Processor holds the minimum and the path identification. These results are transferred off the wafer in word parallel fashion after each input word.

2. Simulations

A detailed simulation of the level-building Dynamic Time Warping (DTW) algorithm has been written for a VAX 11/780. Several features have been included to aid in the analysis of the simulation results. The simulation program includes a complete set of selectable system parameters and design alternatives such as distance metric and path constraint. The documentation capability of the program has been improved by the addition of the ability to produce hardcopy of the graphics display and the inclusion of extensive reporting facilities. Maintaining path accumulation statistics has allowed us to isolate favorable range and magnitude values for the different system configurations. Also, better reporting of actual word and string error statistics provides quantitative information on recognition performance.

The simulation facility has been applied to a subset of both an isolated-word and a connected-word database. These databases include the TI 16-speaker isolated-word database and the TI connected-digit database. Early simulations provided useful directions for further tests and have begun to answer questions in areas where reported results from the literature have been conflicting. Testing will continue until the final hardware design decisions have been made. Results to date show interdependencies among many of the system parameters such as parameter bit quantization and isolated/connected-word speech recognition. In fact, many of the parameter sensitivities do not begin to evidence themselves until they are examined in connected-word context.

Another important observation is that the performance is dependent on details of the front-end processing. This must be investigated so that errors not resulting from the DTW processing can be isolated. The parameters being investigated in the front end include automatic gain control, pre-emphasis, and downsampling of the filter-bank outputs.

A set of five different path constraints has been built into the simulation. Four of these (by Myers, Rabiner, and Rosenberg[7]) were included although they pose various implementation difficulties in the wafer system design. These four were included to provide a basis for comparison.

The fifth, a "flipped" version of the Itakura, was designed to fit the constraints of the row architecture on the wafer. The constraint is suggested in Figure 7. Each node at iteration n + 1 selects the minimum path from three nodes in the previous iteration as shown. The distance computed at node n + 1 is added with different weights depending on which path was chosen. This tends to eliminate the bias that would result from paths passing through differing numbers of nodes.

The simulation also allows the use of several different distance metrics in the node processors. The Chebyshev Norm (L1) and the squared Euclidean have been tested most extensively, since they are the easiest to implement on the wafer. Several others, e.g., log-likelihood, L1/2, L2, and Silverman-Dixon distance calculators, are also available.[8]

The "flipped" Itakura path constraint proposed for use with the row architecture showed a marked improvement over other conventional constraints in isolated word tests. It continues to perform as well as the other constraints, or better, in connected-word tests. This constraint requires only one row of look-back, allowing us to eliminate the need for added storage and a new cell type.

Tests of the different distance metrics were also made. The squared Euclidean distance outperforms the Chebyshev Norm under most conditions. This difference appears to be most marked when automatic gain control is applied.

Tests were run to determine the effect of quantization of the filter-bank parameters for both the input and reference words. The parameter bit-quantization appears to offer negligible improvement in recognition performance above 6 or 7 bits using a Chebyshev Norm, and 5 to 7 bits using the squared Euclidean distance. Six bits will be used in the wafer design.

3. Cell Design

Design of cells for this architecture has begun. Cells are being designed using symbolic layers, with the design rules based on the Lincoln Laboratory 5-μm process then automatically translated to the mask layers for both the MOSIS 3-μm CMOS process and, additionally, the in-house process. This design technique provides enhanced flexibility with respect to the fabrication of the final wafer scale design. The design can be fabricated either in-house or through MOSIS.

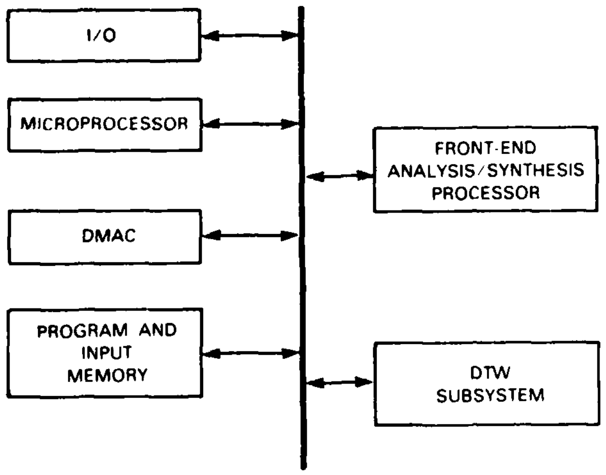
Initially, cells will be fabricated in MOSIS and packaged as discrete chips. This will allow individual cells and small groups of cells to be tested and their designs verified before fabrication of a wafer is started.

Cell design is also being supported by computer simulations. Bit-level simulations of the distance and path computers have been written in the "C" programming language. The simulations will be used to verify and characterize the logical design.

Three partial cell layouts have been completed. The first partial layout, a ROM lookup table for the distance computer, has been fabricated and tested correctly. Two additional partial cell designs have been submitted to MOSIS. These are the two different shift register inputs of the Level Building Processor.

4. DTW System Design

The target application for the DTW wafer is the Advanced Speech Resource Unit (ASRU) being developed under the companion DARPA Wideband Program. This will be a compact peripheral unit containing all the necessary processing facilities to parameterize and recognize continuous speech. Preliminary block diagrams of this system and the associated DTW subsystem are included in Figures 9 and 10, respectively. The system can be described



147995 N

Figure 9. ASRU system block diagram.

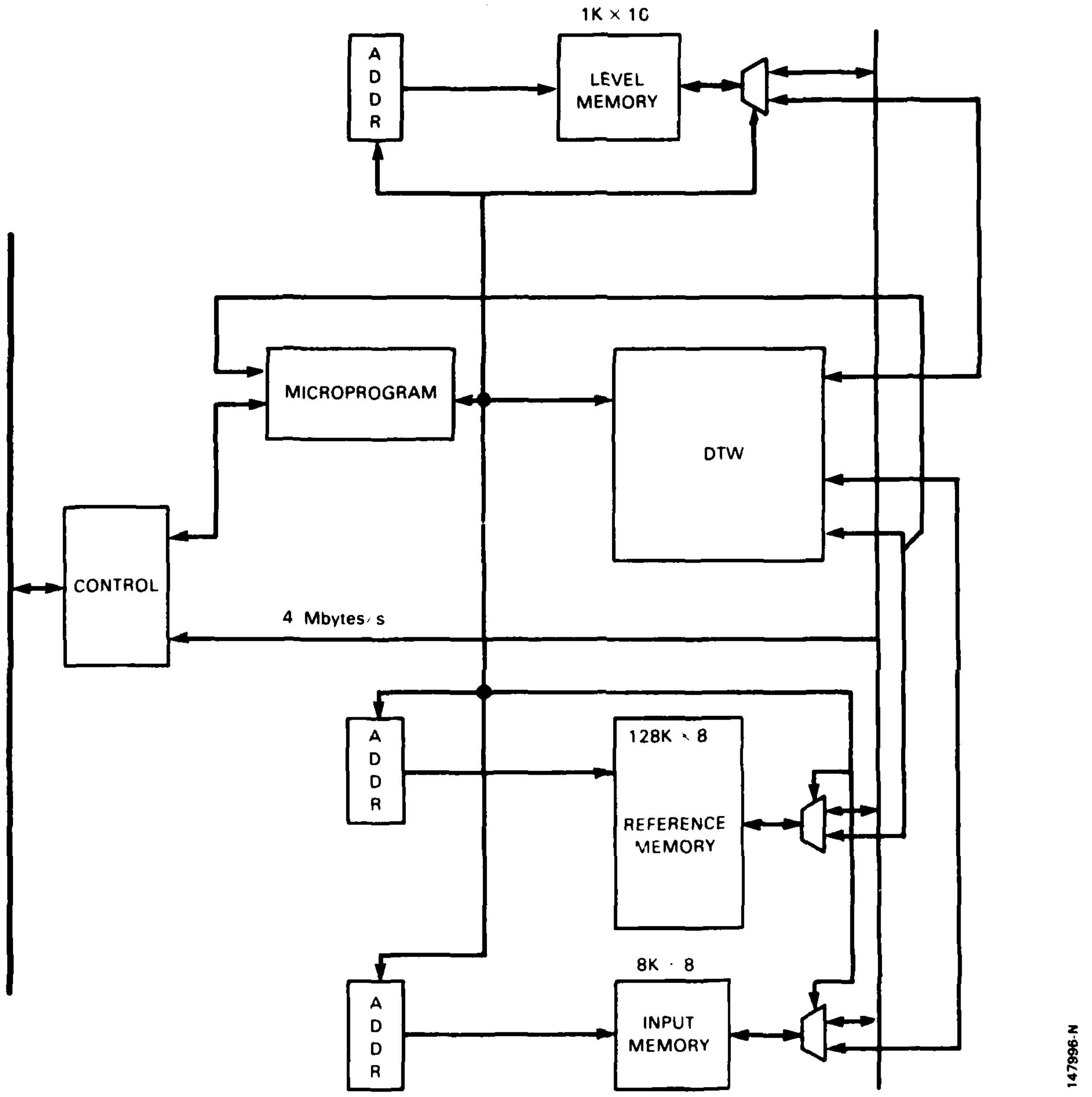


Figure 10. DTW subsystem.

as a general-purpose processor with its associated bus and memory facilities, with analysis/ synthesis and recognizer processors acting as system peripherals. The DTW subsystem is basically a microprogram-controlled data processor, where the microprogram provides both the timing and control signals and handles memory transfers. Development of the ASRU, as well as final architectural decisions, will be the next objective in this program. As architectural questions are answered through simulations, more of the details of the hardware supporting the wafer will be defined.

The resulting design is expected to operate at a clock rate of 4 MHz. Each processing node will be able to complete a distance calculation and path update every 4 μs. The final wafer system, consisting of 60 processing elements, should be capable of processing a 3000 to 5000 reference word vocabulary in real time with a 300-ms response time.

IV. TESTING

A. TOC

A new version of TOC, using new pads modified by hand, was fabricated by MOSIS using the 3-μm NMOS technology. These chips have not yet been tested.

B. OPTICAL PROBE

Shop drawings for the laser source assembly have been completed and a contract awarded to fabricate the parts. The light sources soon will be available.

A number of DARPA contractors have expressed interest in the system. The first units will be provided to the University of Washington, M.I.T., and RADC. A manual covering the theory and operation of the probe and its construction is being prepared.[9]

REFERENCES

1. S. Kirkpatrick, C.D. Gelatt, Jr., and M.P. Vecchi, "Optimization by Simulated Annealing," Science **220**, 671-680 (1983).
2. C.M. Fiduccia and R.M. Mattheyses, "A Linear-Time Heuristic for Improving Network Partitions," Proc. 19th Design Automation Conference, 1982.
3. R.F. Lyon, "Two's Complement Pipeline Multipliers," IEEE Trans. Commun. **COM-24**, 418-425 (1976).
4. S.L. Garverick, "RVLSI FFT Circuits," MIT-Lincoln Laboratory Internal Documentation (February 1983).
5. C.S. Myers and L.R. Rabiner, "Connected Digit Recognition Using a Level-Building DTW Algorithm," IEEE Trans. Acoust., Speech, and Signal Processing **ASSP-29**, 351-363 (1981).
6. H. Murveit, "An Integrated Circuit-Based Speech-Recognition System," Ph.D. Thesis, University of California, Berkeley (14 November 1983).
7. C. Myers, L.R. Rabiner, and A.E. Rosenberg, "Performance Tradeoffs in Dynamic Time Warping Algorithms for Isolated Word Recognition," IEEE Trans. Acoust., Speech, and Signal Processing **ASSP-28**, 623-635 (1980).
8. B.A. Dautrich, L.R. Rabiner, and T.B. Martin, "The Effects of Selected Signal Processing Techniques on the Performance of a Filter-Bank-Based Isolated Word Recognizer," Bell Syst. Tech. J. **62**, 1311-1339 (1983).
9. J.L. Raffel and M.A. Schmidt, "Optical Probe for Integrated Circuit Testing," Project Report RVLSI-8, Lincoln Laboratory, M.I.T. (in press).

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER ESD-TR-84-294	2. GOVT ACCESSION NO. AD-A152 340	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (*and Subtitle*) Restructurable VLSI Program		5. TYPE OF REPORT & PERIOD COVERED Semiannual Technical Summary 1 April — 30 September 1984
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(*s*) Gerald C. O'Leary		8. CONTRACT OR GRANT NUMBER(*s*) F19628-85-C-0002
9. PERFORMING ORGANIZATION NAME AND ADDRESS Lincoln Laboratory, M.I.T. P.O. Box 73 Lexington, MA 02173-0073		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS ARPA Order 3797 Program Element No. 61101E Project No. 3D30
11. CONTROLLING OFFICE NAME AND ADDRESS Defense Advanced Research Projects Agency 1400 Wilson Boulevard Arlington, VA 22209		12. REPORT DATE 30 September 1984
		13. NUMBER OF PAGES 36
14. MONITORING AGENCY NAME & ADDRESS (*if different from Controlling Office*) Electronic Systems Division Hanscom AFB, MA 01731		15. SECURITY CLASS. (*of this report*) Unclassified
		15a. DECLASSIFICATION DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (*of this Report*) Approved for public release; distribution unlimited.		
17. DISTRIBUTION STATEMENT (*of the abstract entered in Block 20, if different from Report*)		
18. SUPPLEMENTARY NOTES None		

19. KEY WORDS (*Continue on reverse side if necessary and identify by block number*)

VLSI	customization	systolic array
Restructurable VLSI (RVLSI)	hardware description language	integrator
programmable interconnect	placement	wafer-scale systems
defect avoidance	routing	speech recognition

20. ABSTRACT (*Continue on reverse side if necessary and identify by block number*)

This report describes work performed on the Restructurable VLSI Program sponsored by the Information Processing Techniques Office of the Defense Advanced Research Projects Agency during the period 1 April through 30 September 1984.

ORIGINATOR - SUPPLIED KEY WORDS INCLUDE: